中國機械工程學刊第二十一卷第一期第 67~76 頁（民國八十九年）
Journal of the Chinese Society of Mechanical Engineers, Vol.21, No.1, pp.67~76 (2000)

Shape and Vibration Control of a Multibody Flexible Structure Using Recurrent Neural Networks

Franco Bernelli-Zazzera*, Amalia Ercoli-Finzi**, Marcello Romano*** and Massimo Tomasi****

Keywords: neural control, flexible structures, multibody systems.

ABSTRACT

This paper deals with the active control of a space multibody system, composed by a two link rigid manipulator and a large flexible truss. The manipulator is a two-link robot with all its parts completely rigid and three joints for its motion. The truss is a typical large space truss with low frequency vibration modes. In particular the truss shape control and manipulator positioning control are investigated using recurrent neural networks. Autonomous controllers are designed for each robot joint to control the manipulator position and movement along the truss and for the truss to damp the vibrations induced by the robot motion. Recurrent networks allow to design adaptive controllers, capable to cope with the non linear, time varying dynamics produced by the robot motion. The use of recurrent neural controllers for this application reduces the effort in the system modeling. In fact, the controllers are able to adapt to time varying situations, so even the changes in inertial characteristics due to the relative motion between truss and manipulator are automatically taken into account in a transparent way, once a suitable structure of the recurrent networks has been designed.

INTRODUCTION

Paper Received September, 1999. Revised November 1999. Accepted December, 1999. Author for Correspondence: Franco Bernelli-Zazzera.

* *Associate Professor, Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, Via La Masa 34- 20158 Milano, Italy.*

** *Professor, Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, Via La Masa 34- 20158 Milano, Italy.*

*** *Ph.D. Candidate, Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, Via La Masa 34- 20158 Milano, Italy.*

**** *Graduate Student, Dipartimento di Ingegneria Aerospaziale, Politecnico di Milano, Via La Masa 34-20158 Milano, Italy.*

The huge dimension of the International Space Station (ISS) introduces new problems never dealt with on previous orbital bases or other space systems.
Structures like ISS are named Large Space Structures (LSS). One of the most important characteristics of a LSS is to have low frequency vibration modes. Internal disturbances (like crew movements) and external disturbances (e.g., due to interactions with a docked shuttle) produce oscillations on flexible parts that stay on for a long time.
The complete analysis of this kind of systems requires a wide investigation: it ranges from the dynamics, identification and control strategies, position control and reaction forces among parts.
In particular interaction between two bodies, one of them flexible, with relative motion was studied in a limited situation. Most papers are relative to attitude dynamic coupling with rigid and flexible parts. Just few papers deal with shape control and relative movement among bodies.
Silverberg and Park (1990) developed a theoretical analysis showing the relevant interactions among rigid and flexible bodies during space maneuvers. Chan and Modi (1989) presented a Lagrange formulation to study the attitude of a flexible base with two rigid appendices. Bennett et al. (1994) tested a non-linear control strategy based on Partial feedback linearization (PFL).
The study developed by Messac (1996) is closer to the work presented in this paper. The author developed an analytical approach limited to the dynamic behavior of a time-varying system constituted by a trolley moving on a beam. The system equations of motion were built independently and then joined together in one system to be solved with a classical method.
Yangsheng and Ueno (1994) analyzed the model of a space manipulator with five degrees of freedom with a

gravity compensation control system to simulate zero gravity conditions. He developed both a rigid and flexible model and made experimental tests.
Bernelli-Zazzera and Ercoli-Finzi (1996) verified the behavior of a system constituted by a flexible truss and a manipulator with two links. The truss dynamics appears strongly connected to the manipulator dynamics since the low frequency vibration modes of the structure are close to the manipulator frequencies.
The present paper too is focused on the control of the dynamics of a space system composed by a manipulator and a truss with low frequency vibration modes. In particular, starting from previous experimental works (Bernelli-Zazzera and Lo-Rizzo, 1997, Bernelli-Zazzera and Lo-Rizzo, 1999), we tested, through numerical simulations, the adaptability of a recurrent neural oscillation control to the relevant variations of inertial characteristics, induced by the movement of the manipulator on the truss. A position control of the manipulator has been furthermore realized, independent from the control of the truss, but using the same neural architecture. The main issue becomes then to structure the recurrent networks in such a way to allow them to "learn" the system behavior faster than the robot motion. Once this effort is completed, the neural networks can properly control the time varying system without any need to model the structure of the time dependent dynamic interactions between truss and manipulator. This feature represents a great simplification in system modeling, although the range of validity of the proposed approach has not been theoretically demonstrated yet. In the present application, the robot has successfully completed a maneuver along the entire span of the truss.
The paper is organized as follows: first a detailed description of the space system used for the simulations will be reported, followed by an introduction to the methodology used for the identification and control, then the identification and control strategy applied to the truss and to the manipulator will be presented and finally the simulation results will be discussed.

DESCRIPTION OF THE MULTIBODY SYSTEM

The multibody system considered is composed by two main subsystems: a 3 degrees of freedom planar manipulator with two rigid links and a flexible structure with truss topology.

The Truss

The truss considered for simulations corresponds to a real reticular structure named Truss Experiment for Space Structures (TESS), set up in the laboratory of the Aerospace Engineering Department of Politecnico di Milano (Bernelli-Zazzera and Lo-Rizzo, 1997). It is a modular structure with longitudinal growth and square section (Fig.1).

Table 1. Physical characteristics of the truss.

Mass [Kg]	I_{xx} [Kgm^2]	I_{yy} [Kgm^2]	I_{zz} [Kgm^2]	I_{xy} [Kgm^2]	I_{xz} [Kgm^2]	X_{CM} [m]	Y_{CM} [m]	Z_{CM} [m]
80.63	5.039	9806	9806	-0.440	-0.648	9.531	-5.95E-4	-5.95E-4

Fig.1. Scheme of the truss TESS.

The complete length of the structure is 19.062 meters. The truss is composed of 54 cubic modules (Fig.2). Each module has section size of $0.353m$ and spherical joints and tube rods, made of PVC, constitute it. Table 1 reports the main physical characteristics of TESS.
For simulations the structure is considered completely free to move in space, without gravity and constraints. The finite element model used for the simulations has already been validated by a large number of experimental tests. That FE model is here taken in account with the exclusion of the six constraining springs used to sustain the truss on the laboratory ceiling (Bernelli-Zazzera and Lo-Rizzo, 1997).
The first non-rigid modes, which are bending modes, are reported in Table 2.

The Manipulator

The manipulator used for the simulation is a three rotational degrees of freedom planar robot with two links (Fig.3). The links have a distributed mass along a line between joints. The overall joint systems, comprehensive of torque motors and connection elements, are modeled as lumped masses placed on the axes of the joints. Table 3 reports all inertial and material characteristics. All parts of the manipulator are considered rigid.
The geometrical lengths and inertial characteristics have been chosen to be realistic. In this way these simulation results and the identification and control parameters could be used as starting point for a possible future experimentation.

Sensors and Actuators

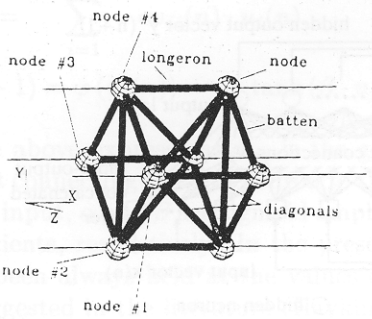

Fig.2. Scheme of one module of the TESS truss.

Table 2. First bending frequencies.

Number	First bending frequencies	Mode shape
1,2	1.085 (Hz)	Horizontal and vertical
3,4	3.042 (Hz)	Horizontal and vertical
5,6	5.760 (Hz)	Horizontal and vertical
7,8	9.317 (Hz)	Horizontal and vertical

In the simulations, sensors and actuators are virtual systems, respectively rotation and angular velocity sensors for the manipulator and accelerometers and thrusters with continuously variable thrust for the truss. The data that in reality would be measured by sensors are instead obtained by integration of the dynamic equations of motion.
The truss has 4 accelerometers and 8 thrusters, located at sections 1,24,32,55 of the truss. The choice of the location of sensors and actuators is based on observability and controllability analyses focused on the first four bending modes. Both sensors and actuators are placed in the points with maximum modal participation, obtained from a model with the previous indicated four vibration modes. The actuators are simply modeled using forces and torques.

Maneuvers

The manipulator moves along an edge of the truss locking and unlocking the end-effector on truss nodes. Two different kinds of maneuvers have been simulated: the "acrobat maneuver" and the "grub maneuver". While the grub maneuver implies a sliding movement of the robot along the truss, the acrobat maneuver carries out an overturning movement. Then the second one generates a much higher level of disturbance on the truss dynamics and higher moment of inertia variations and is more interesting from a control system testing point of view.

The Acrobat Maneuver

The manipulator displacement during this maneuver

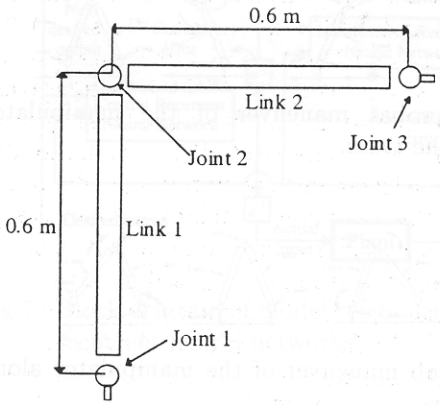

Fig.3. Manipulator scheme.

Table 3. Physical characteristics of the manipulator.

Inertial Characteristics	Link 1 = Link2
Mass	0.39 kg
I_{xx}	5.1617 E-3 kg m^2
I_{yy}	1.3166 E-4 kg m^2
I_{zz}	5.1617 E-3 kg m^2
Material Characteristics	Aluminum
Young Modulus	7.170 E+10 kg m^{-2}
Poisson Coefficient	0.33
Density	2740 kg m^3

is described by the image sequence in Fig.4. Considering Fig.4.a as starting configuration, once the joint 1 is unlocked the movement begins and joints 2 and 3 rotate in a negative verse. Link 1 rotates of 5.025 radians with respect to joint 2 while link 2 must rotate of 1.258 radians with respect to joint 3. The operation continues until the end-effector on joint 1 reaches the contact point at truss node E and is locked again. That end position is two nodes ahead of the end-effector starting point.

The Grub Maneuver

In this case the maneuver does not involve any overturning of the links. The maneuver starts by unlocking the end-effector relative to joint 1 and then rotating link 1 with a positive value of 0.33 radians with respect to joint 2, while link 2 rotates of a same but negative value with respect to joint 3. When the joint 1 reaches point B the first part of the maneuver ends (Fig.5.b). The second step consists in unlocking the joint 2 and move it two nodes ahead with respect

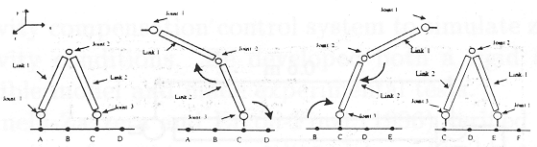

Fig.4. The acrobat maneuver of the manipulator along the truss.

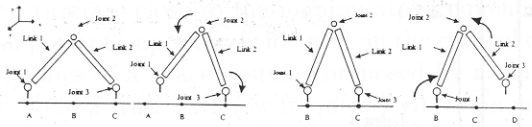

Fig.5. The grub maneuver of the manipulator along the truss.

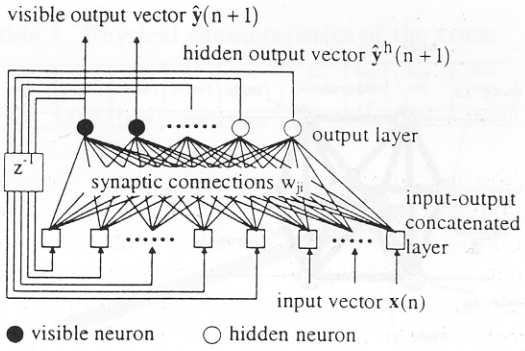

Fig.6. Fully connected recurrent network.

to the end-effector joined to the structure. Link 1 rotates of -0.33 radians on joint 1; link 2 rotates of 0.33 radians on joint 2.

RECURRENT NEURAL NETWORKS

The basis of any neural control method is represented by the emulation of the inverse dynamics of the physical system to be controlled. This can be achieved either by direct or indirect modeling.

In the direct modeling inversion, a neural network is trained using as input the measured output of the physical system. The network output represents the input that presumably produced the system output, and its difference with the real input to the system drives the learning algorithm.

In the indirect modeling inversion instead, the control network is trained using as input the desired output of the physical system. The output of the net is the input to the physical system, and the difference between the actual and desired system response drives the learning algorithm. In some cases, the actual system output is replaced, in the learning algorithm, by the output of an identification net, fed with the same input as the actual system. This scheme (Morelli, 1992) is very convenient since it allows an implicit filtering of the data and a straightforward development of the learning algorithm, requiring absolutely no knowledge of the mathematical model of the dynamics of the physical system. So, even if the system is non-linear and time varying, an adaptive control based on indirect inversion may be applicable, provided that the identification net is able to learn faster than the control net.

The choice of the topology of the identification and control networks must take into account the necessity of updating all the synaptic weights during each sampling period. Use of the popular multi-layer perceptron topologies implies a great number of inputs, neurons and connections, since a static neural-net has to monitor current and past measurements. Inevitably, such a solution makes the computation of all synaptic weights variations during a short sampling period impossible. Unfortunately short sampling periods are imposed by the necessity of avoiding aliasing problems. The simplest and most effective way to identify a dynamic system using ANNs is then to apply intrinsically dynamic networks. Among these, the most compact are indeed recurrent networks (RN) (Haykin, 1994).

A typical fully connected RN is represented in Fig.6. It consists of a layer of neurons whose one-step delayed outputs combined with the external input form the concatenated input-output layer.

Visible neurons, those giving the predicted output vector $\hat{\mathbf{y}}(n+1)$, are used to compute the weight updates while the remaining act as hidden neurons. Joining the system input and output forms the concatenated input-output layer. As will be detailed in the sequel of the presentation, the system input can also include, besides the physical inputs (forces), extra quantities useful to the prediction of the output vector.

Calling $\mathbf{x}$ the vector of ni system inputs, $\hat{\mathbf{y}}$ and $\hat{\mathbf{y}}^h$ the vectors of no visible and nho hidden outputs of the network, then at each sampling instant n, the concatenated input-output layer can be expressed in vector notation as

$$\mathbf{u}(n) = \left[\mathbf{x}(n)^T \ \hat{\mathbf{y}}(n)^T \ \hat{\mathbf{y}}^h(n)^T\right]^T. \qquad (1)$$

The network dynamics is defined by the net internal activity level v_j related to each j-th neuron and by its sigmoidal activation function φ which, in this paper, has been characterized with a hyperbolic tangent

$$v_j(n) = \sum_{i=1}^{ni+no+nho} w_{ji}(n) \cdot u_i(n),$$

$$\hat{y}_j(n+1) = \varphi(v_j(n)) = \alpha \cdot \tanh(\beta \cdot v_j(n)). \qquad (2)$$

In the above expressions w_{ji} is the generic synaptic weight connecting the j-th neuron to the i-th concatenated input, α and β are sigmoid amplitude and slope coefficients, respectively. In the present work, these have been always held at the values of $\sqrt{3}$ and 2/3, as suggested in the literature (Haykin, 1994, Guyon, 1991).

Williams and Zisper (1989) and Williams and Peng (1990) derived a learning algorithm for RNs from Extended Back Propagation (EBP). It is based on the minimization of the error cost function

$$E(n) = \frac{1}{2}\mathbf{e}(n)^T \cdot \mathbf{e}(n), \qquad (3)$$

where $\mathbf{e}$ is the innovation, difference between measured and one step delayed predicted visible output

$$\mathbf{e}(n) = \mathbf{y}(n) - \hat{\mathbf{y}}(n). \qquad (4)$$

At each sampling instant n, the synaptic weight variation can then be obtained by the steepest descent method, also called delta rule. The weight matrix is updated according to the relation

$$\Delta w_{k1}(n) = -\eta \frac{\partial E(n)}{\partial w_{k1}(n)},$$

$$w_{k1}(n+1) = w_{k1}(n) + \Delta w_{k1}(n), \qquad (5)$$

where η represents the learning rate of the network. The indexes k and 1 range from 1 to the number of neurons and from 1 to the number of concatenated inputs, respectively. Recalling Eqs. (2)-(4), the last factor of Eq.(5) can be calculated at each iteration starting from its own previous value, from past inputs and weights, as detailed in the Appendix.

IDENTIFICATION AND CONTROL STRATEGY

Actual Topologies and Connections Among Networks

The use of neural controllers for this application has the advantage of reducing the effort in the system modeling. The use of Recurrent Neural Networks, instead of more popular static neural-net, makes the computation of all synaptic weights variations during a short sampling period possible.

In the present application network topologies have been slightly modified with respect to the classical recurrent scheme. Two fundamental reasons guided these modifications. First, the computational time must not act as a delay in the application of the desired input to the system. So the networks are designed as predictive networks, and the proper input to the system is evaluated with one sampling period of advance. In a real application this allows for computational times about as long as the sampling time. The second motivation for the network modification resides in the lack of a supervisor from which the control net learns to behave. So, the control net must perform its training process autonomously, on the basis of the plant inputs and outputs. To make this possible, system identification must be completed before the activation of the control network. As depicted in Fig.7, the overall system consists therefore of the identification network working in parallel with the actual structure, both driven by the control network working with one sampling period of advance. Figures 8 and 9 represent a detailed layout of the two networks. The identification net has *nho* hidden and *no* visible neurons. The information about actual plant dynamics comes directly from the set of

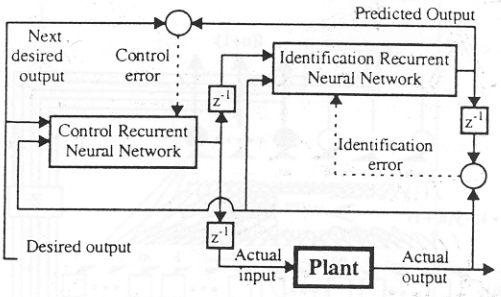

Fig.7. Block diagram of model reference control using recurrent neural networks.

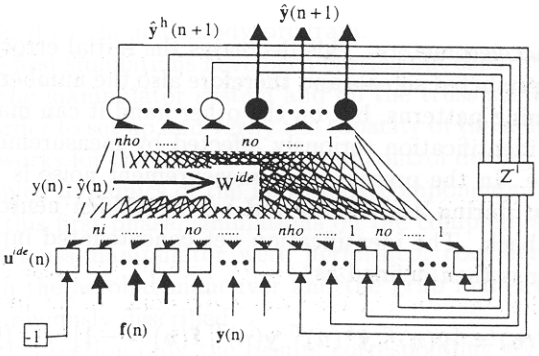

Fig.8. Identification recurrent neural network.

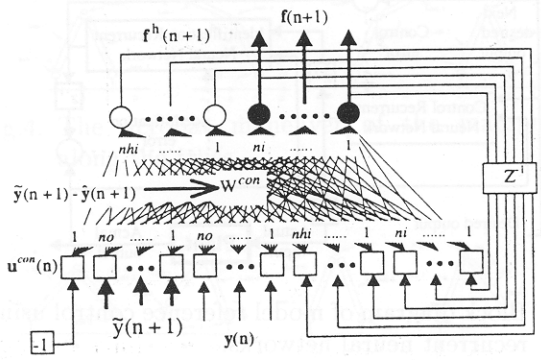

Fig.9. Control recurrent neural network.

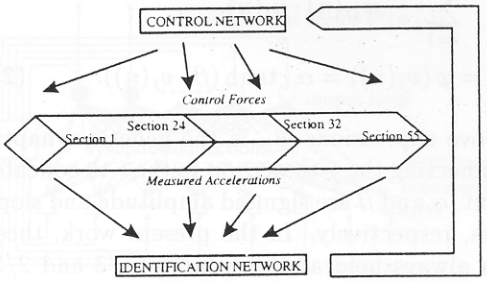

Fig.10. Truss control system scheme.

no accelerometers. This decreases the initial error of the predicted signals and therefore also the number of training patterns, but on the other hand it can make the identification seriously affected by measurement noise. In the present case, measurement noise is filtered during acquisition and by the hidden neurons feedback. The identification net concatenated input layer is then defined as

$$\mathbf{u}^{ide}(n) = \left[\hat{\mathbf{y}}(n)^T\ \hat{\mathbf{y}}^h(n)^T\ \mathbf{y}(n)^T\ \mathbf{f}(n)^T\ -1\right]^T, \quad (6)$$

where $\hat{\mathbf{y}}(n)$ is the one step delayed predicted output vector, $\hat{\mathbf{y}}^h(n)$ is the one step delayed predicted output vector belonging to hidden neurons, $\mathbf{y}(n)$ is the actual plant output vector (the measured accelerations), $\mathbf{f}(n)$ represents the command coming from the control neural net and -1 drives the neuron threshold via its related synaptic weights. For the identification net, the weight variation is still computed starting from the prediction error defined as difference between the actual output and the predicted output belonging to visible neurons.
On the other hand, the control net is fed with concatenated inputs defined by the following expression

$$\mathbf{u}^{con}(n) = \left[\mathbf{f}(n)^T\ \mathbf{f}^h(n)^T\ \mathbf{y}(n)^T\ \tilde{\mathbf{y}}(n+1)^T\ -1\right]^T, \quad (7)$$

where the predicted output of visible neurons, $\mathbf{f}(n+1)$, is delayed and fed back to the net; it represents the control command to be furnished to the physical plant and to the identification net at the next step. Other inputs to the control network are the actual plant outputs $\mathbf{y}(n)$, the desired output at the next acquisition $\tilde{\mathbf{y}}(n+1)$, coming from a reference model, and the threshold driver -1.
Control weights update is now based on the error between desired and predicted visible output. No presence of physical plant measurements denotes that updating control neural weights is conceptually slightly different from the modified delta rule mentioned above. To face uncertainties deriving from time varying dynamics, the behavior of the physical system, crucial for the determination of the proper control action, is recursively captured from the recurrent identification neural network working in parallel with the plant (see Fig.7). This approach modifies the control learning algorithm, since it must include the interaction with the identification net. The complete learning algorithms of the identification and control networks are collected in the Appendix.
It is pointed out that the performance of the control net depends on the accuracy of the identification. It is remarked that this is due to the lack of a supervisor from which the control RN learns its behavior. The training process is autonomously realized by the control network, which monitors the plant input and output and manipulates control input to make the predicted output, based on the identified model of the system, equal to the desired output. To make this process possible, system identification must be already completed when the control system begins to operate to adjust its weights.

Truss Subsystem

The identification network identifies the truss system using the measured four accelerations, i.e. it gives a prediction on acceleration corresponding to input forces; the control network, considering the acceleration values predicted and wanted, in this case zero, gives, as output, the magnitude of actuator forces (Fig.10). The Learning phase is based on exciting the structure with periodic forces having the same spectrum width of the considered vibration modes of the truss.
The synaptic weights of the identification network are updated minimizing the error due to the difference between the signal of sensors and the output of the

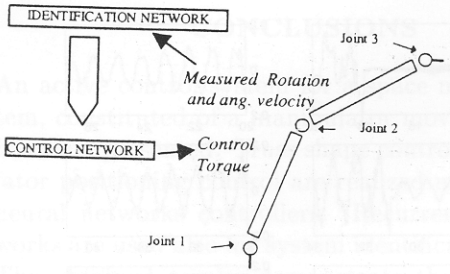

Fig.11. Control system for one robot joint.

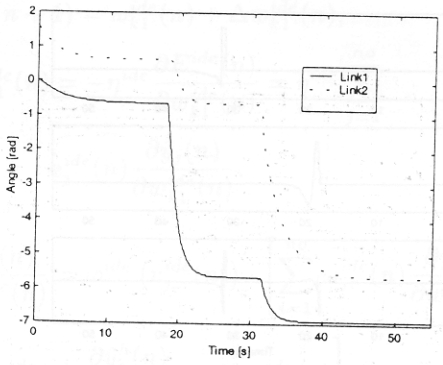

Fig.12. Angle displacements of the two links during one step of the acrobat maneuver.

identification net itself, delayed one step. Instead, the synaptic weights of the control net are updated minimizing the distance between the desired final value of acceleration, i.e. zero, and the predicted value produced by the identification network. With reference to Figs. 8 and 9, the neural nets are defined by $no = 4$, $nho = 6$, $ni = 4$, $nhi = 6$.

Manipulator Subsystem

The same recursive neural network control architecture has been used for the positioning of each joint of the robot, resulting in three independent systems for the robot motion control.
Identification inputs are the measured joint angular position and velocity. The control network produces the torque values that the electrical-motors must apply to the manipulator arms to obtain the wanted final configuration of the robot (Fig.11).
The learning cost function for the control network is the sum of two errors: the first is the difference between current and desired position; the second is the distance between the current and desired velocity. In this case, for each joint $no = 2$, $nho = 4$, $ni = 1$, $nhi = 4$. Starting weights for control system of joint 1 and 3 are the same. To obtain them, in the learning phase, a known motion to joint 1 with joint 2 fixed was applied. Instead, for joint 2, a known motion has been applied with joint 1 fixed.

SIMULATION RESULTS

Simulations have been performed using routines written with the commercial software for multibody systems ADAMS. Those ADAMS routines were linked to the routines implementing the neural networks, written in FORTRAN. The flexibility of the truss was considered by introducing the vibration modes as evaluated by NASTRAN, and coupling them to the rigid parts directly modeled by ADAMS. The actuators are simply modeled using forces and torques defined by the multibody program.
The first simulations have been carried out separately on the manipulator system and on the truss system, in order to set up and verify the capacity of the neural networks for the identification and control during decoupled dynamics of the multibody components. After this first phase, simulations on the complete system truss-manipulator were performed, considering both the acrobat maneuver and the grub maneuver, as previously described.
In this section only the results corresponding to the acrobat maneuver of the manipulator moving on the truss will be presented. This because that maneuver induces on the truss acceleration perturbations almost one order of magnitude greater than the grub maneuver, so it is more interesting to analyze from the point of view of the interactions between the manipulator system and the truss system. The results presented will be those relative to one step of the maneuver, since this is representative of the typical behavior of the system as the manipulator moves from one end of the truss to the opposite one. Figure 12 reports the angle displacements of the manipulator links during the first step of the maneuver. At time 0 the manipulator is placed at one extremity of the truss with joint 1 locked on it, link1 disposed normal to the truss axis (angle displacement $= 0$) and link 2 parallel to the truss axis (angle displacement $= \pi/2$). A first phase of maneuver (between 0 and 18 s) brings the manipulator to the typical starting configuration of acrobat maneuver (see Fig.4.a). Then a first overturning, between 18 and approximately 32 s, and a second overturning, after 32 s, brings the manipulator to the final rest configuration. The truss control is switched on at time equal to 18 s.
Figure 13 reports the control torques applied to the manipulator joints to execute the acrobat maneuver. An interactive simulation achieves the connec-

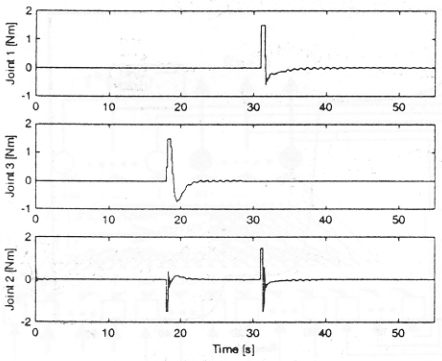

Fig.13. Torques applied by the NN controllers to robot joints during acrobat maneuver.

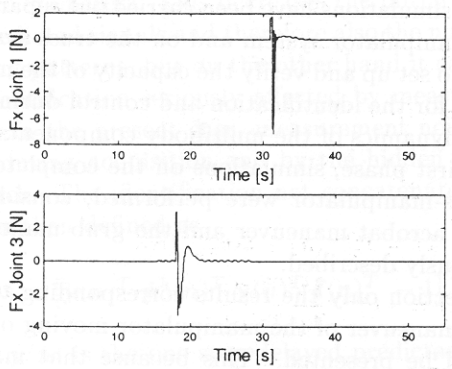

Fig.14. Forces applied by the robot joints locked to the truss during acrobat maneuver.

tion and disconnection of the manipulator joints with the truss. When a virtual proximity sensor infers that the distance between the robot end effector and the target node on the truss gets lower than $1mm$, the simulation is stopped. At this point one end effector is locked to the node and the other unlocked using a special tool offered by ADAMS. Then the simulation is started again.

Each time a joint is locked to a truss node the control is stopped for two seconds to avoid integration problems due to the great variation of dynamic quantity. In this transition phase the neural networks are disabled to avoid identification and control instability caused by the low input values.

Figure 14 shows a typical example of how the forces are transmitted by the manipulator to the truss. Each time a new rotation of one link of the manipulator is commanded, a peak in the truss reaction force appears. This produces obviously a peak in the truss acceleration, as will be shown further on, but with no consequences on the global performances of either

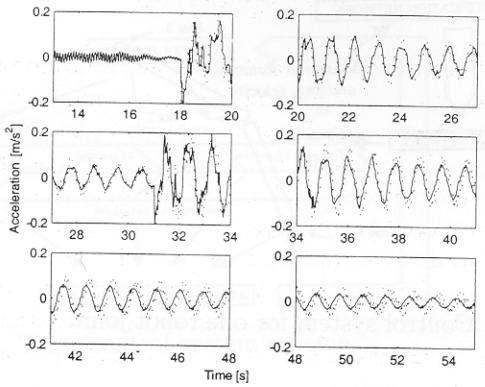

Fig.15. Output of the accelerometer at the extremity of the truss during the maneuver (solid line: with control; dashed line: without control).

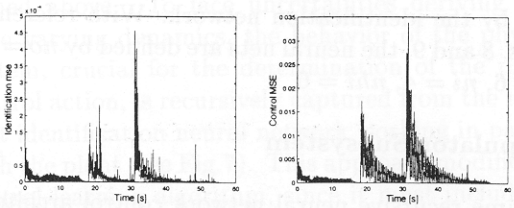

Fig.16. Mean square errors of the truss identification and truss control neural networks.

truss or manipulator, which can complete the maneuver safely and precisely. This is mainly due to the fact that all the neural networks have been trained correctly to withstand this level of disturbances.

Figure 15 reports the acceleration measured by the sensor mounted at the extremity of the truss, where the maneuver begins; the acceleration trend measured by the other three sensors are similar to the former one. The synaptic weights of both the identification and the control networks are initialized, before the maneuver, using the values saved at the end of the set up tests, carried out separately on the manipulator and the truss. The presence of the control makes the acceleration level about 50% lower with respect to the case without control.

Figure 16 represents the errors driving the learning algorithms of the truss identification and control networks. It is possible to notice that, each time a new overturning phase of the maneuver begins, there is a learning period for the neural networks. During those periods the mean square errors rise, then they rapidly decrease again. Moreover the learning process of the identification net results faster than the equivalent process for the control net, as it should be.

CONCLUSIONS

An active control system for a space multibody system, constituted of a manipulator moving on a truss has been presented. Truss shape control and manipulator positioning control are realized using recurrent neural networks controllers. Recurrent neural networks are used also for system identification purpose. The obtained results demonstrate the good adaptability of the neural architecture oscillation control to the relevant variations of the inertial characteristics, induced by the movement of the manipulator on the truss.
During the maneuver the identification and control networks are started up with the final synaptic weights of previous training phases, operated separately on the manipulator and truss subsystems. Then networks become able to identify and control the overall system.

ACKNOWLEDGMENTS

ASI (Italian Space Agency) has supported this research under contract ASI ARS-96-127.

APPENDIX

Learning Algorithms for the Identification and Control Networks

In the following, the superscripts *ide* and *con* refer the related parameters to identification and control networks respectively, *no* and *nho* indicate the number of visible and hidden neurons of the identification network, *ni* and *nhi* are the number of visible and hidden neurons of the control network. The vector $\mathbf{u}$ is the concatenated input to the network, v is the neural activity, E is the error function to be minimized, α and β define the neuron function, φ' is the first derivative of the sigmoidal activation function and δ is the Kroneker delta.

$$v_j^{ide}(n) = \sum_{i=1}^{ni+2no+nho+1} w_{ji}^{ide}(n) \cdot u_i^{ide}(n),$$

$$\hat{y}_j(n+1) = \varphi^{ide}\left(v_j^{ide}(n)\right) = \alpha^{ide} \cdot \tanh\left(\beta^{ide} \cdot v_j^{ide}(n)\right), \tag{9}$$

$$\mathbf{e}^{ide}(n) = \mathbf{y}(n) - \hat{\mathbf{y}}(n), \quad E^{ide}(n) = \frac{1}{2}\mathbf{e}^{ide}(n)^T \cdot \mathbf{e}^{ide}(n), \tag{10}$$

$$w_{k1}^{ide}(n+1) = w_{k1}^{ide}(n) + \Delta w_{k1}^{ide}(n),$$

$$\Delta w_{k1}^{ide}(n) = -\eta^{ide}\frac{\partial E^{ide}(n)}{\partial w_{k1}^{ide}(n)} = \eta^{ide}\sum_{j=1}^{no} \mathbf{e}_j^{ide}(n)\frac{\partial \hat{y}_j(n)}{\partial w_{k1}^{ide}(n)}, \tag{11}$$

$$\frac{\partial \hat{y}_j(n)}{\partial w_{k1}^{ide}(n)} = \dot{\varphi}^{ide}\left(v_j^{ide}(n)\right) \cdot \left[\sum_{i=1}^{no} w_{ji}^{ide}(n)\frac{\partial \hat{y}_i(n)}{\partial w_{k1}^{ide}(n)} + \sum_{i=1}^{nho} w_{ji}^{ide}(n)\frac{\partial \hat{y}_i^h(n)}{\partial w_{k1}^{ide}(n)} + \delta_{kj}u_1^{ide}(n)\right], \tag{12}$$

$$v_r^{con}(n) = \sum_{s=1}^{2no+ni+nhi+1} w_{rs}^{con}(n) \cdot u_s^{con}(n),$$

$$f_r(n+1) = \varphi^{con}\left(v_r^{con}(n)\right) = \alpha^{con} \cdot \tanh\left(\beta^{con} \cdot v_r^{con}(n)\right), \tag{13}$$

$$\mathbf{e}^{con}(n) = \tilde{\mathbf{y}}(n+1) - \hat{\mathbf{y}}(n+1),$$

$$E^{con}(n) = \frac{1}{2}\mathbf{e}^{con}(n)^T \cdot \mathbf{e}^{con}(n), \tag{14}$$

$$w_{tv}^{con}(n+1) = w_{tv}^{con}(n) + \Delta w_{tv}^{con}(n),$$

$$\Delta w_{tv}^{con}(n) = -\eta^{con}\frac{\partial E^{con}(n)}{\partial w_{tv}^{con}(n)} = \eta\sum_{j=1}^{no} \mathbf{e}_j^{con}(n)\frac{\partial \hat{y}_j(n+1)}{\partial w_{tv}^{con}(n)}, \tag{15}$$

$$\frac{\partial \hat{y}_j(n+1)}{\partial w_{tv}^{con}(n)} = \dot{\varphi}^{ide}\left(v_j^{ide}(n)\right) \cdot \sum_{r=1}^{ni} w_{j(r+2no+nho)}^{ide}(n) \cdot \frac{\partial f_r(n)}{\partial w_{tv}^{con}(n)}, \tag{16}$$

$$\frac{\partial f_r(n+1)}{\partial w_{tv}^{con}(n)} = \dot{\varphi}^{con}\left(v_r^{con}(n)\right) \cdot \left[\sum_{s=1}^{ni} w_{rs}^{con}(n) \cdot \frac{\partial f_r(n)}{\partial w_{tv}^{con}(n)} + \sum_{s=1}^{nhi} w_{rs}^{con}(n) \cdot \frac{\partial f_r^h(n)}{\partial w_{tv}^{con}(n)} + \delta_{tr} \cdot u_v^{con}(n)\right]. \tag{17}$$

REFERENCES

Bennett, W.H., Kwatny, H.G., and Beak, M.J., "Nonlinear Control of Flexible, Articulated Spacecraft: Application to Space Station/mobile Manipulator," *Journal of Guidance, Control, and Dynamics,* Vol.1, No.17, pp.38-47 (1994).

Bernelli-Zazzera, F., and Ercoli-Finzi, A., "Maneu-

vers of Flexible Space Manipulator," *Applied Mechanics in Americas,* Vol.5, pp.403-406 (1996).

Bernelli-Zazzera, F., and Lo-Rizzo, V., "Adaptive Control of Space Structures: An Experiment Based on Recurrent Neural Networks," *Proceedings of the Eleventh VPI&SU Symposium on Structural Dynamics and Control,* Blacksburg, Virginia, pp.605-614 (1997).

Bernelli-Zazzera, F., and Lo-Rizzo, V., "Adaptive Control of Space Structures via Recurrent Neural Networks," *Dynamics and Control,* Vol.9, No.1, pp.5-20 (1999).

Chan, J.K., and Modi, V.J., "A Formulation for Studying Dynamics of An Orbiting Flexible Mobile Manipulator," *Proceedings of the Seventh VPI&SU Symposium on Dynamics and Control of Large Structures,* pp.492-506 (1989).

Guyon, I., "Applications of Neural Networks to Character Recognition," *International Journal of Pattern Recognition and Artificial Intelligence,* Vol.5, pp.353-382 (1991).

Haykin, S., *Neural Networks: A Comprehensive Foundation,* Maxwell Macmillan International, (1994).

Messac, A., "Flexible-Body Dynamics Modeling of A Vehicle Moving on the Rails of A Structure," *Journal of Guidance, Control, and Dynamics,* Vol.19, No.3, pp.540-548 (1996).

Morelli, G., "Applications of Neural Networks to the Adaptive Control of Spacecraft," *Estec Working Paper* No.1651, (1992).

Silverberg, L.M. and Park, S., "Interactions Between Rigid-Body And, Flexible-Body Motions in Maneuvering Spacecraft," *Journal of Guidance, Control, and Dynamics,* Vol.13, No.1, pp.73-81 (1990).

Williams, R.J., and Zisper, D., "A Learning Algorithm for Continually Running Fully Recurrent Neural Networks," *Neural Computation,* Vol.1, pp.270-280 (1989).

Williams, R.J., and Peng, J., "An Efficient Gradient Based Algorithm for On-Line Training of Recurrent Network Trajectories," *Neural Computation,* Vol.2, pp.490-501 (1990).

Yangsheng, Xu Y., and Ueno, H., "Modelling and Configuration — Independent Control of A Self — Mobile Space Manipulator," *Journal of Intelligent and Robotics Systems,* Vol.10 (1994).